Theoretical Computer Science 132 (1994) 71–84
Elsevier

Unrestricted complementation in language equations over a one-letter alphabet*

E.L. Leiss

Department of Computer Science, University of Houston, Houston, TX 77204-3475, USA

Communicated by M. Nivat
Received August 1992
Revised October 1993

Abstract

Leiss, E.L., Unrestricted complementation in language equations over a one-letter alphabet, Theoretical Computer Science 132 (1994) 71–84.

Recently, the complete solution of systems of language equations over a one-letter alphabet has been derived where the operators are union, concatenation and star. This paper addresses the question of adding the complementation operator. It is known that if no stars are present and if the concatenation operator is restricted, such equations need not have solutions, but if they do, they can be solved explicitly if the constants are regular in which case the solutions are also regular. In this paper we study language equations over a one-letter alphabet where unrestricted concatenation interacts with complementation, union, and star. We define a procedure for determining a regular solution of the general type of equation assuming that all constants are regular, discuss its properties, and show examples. We then prove that in general equations with complementation and concatenation in which all constants are regular may have nonregular solutions. This shows that language equations can be used to obtain non-context-free languages from regular languages using only operators under which the regular languages are closed.

1. Introduction and notation

A language equation in the variable over the alphabet A is defined as follows:

$$X = \alpha,$$

Correspondence to: E.L. Leiss, Department of Computer Science, University of Houston, Houston, TX 77204-3475, U.S.A. Email: coscel@cs.uh.edu.

* Part of this research was carried out while the author was visiting the Universidade Federal do Rio Grande do Sul in Porto Alegre, Brazil, supported by a grant from CNPq.

SSDI 0304-3975(93)E0197-C

where α (or $\alpha(X)$ if we want to put the variable X explicitly in evidence) is a member of the class $\mathrm{REG}_A(X)$ of regular expressions in the variable X over A. $\mathrm{REG}_A(X)$ is defined as follows:

(1) A language L over A is in $\mathrm{REG}_A(X)$; L is called a constant (language). The variable X is in $\mathrm{REG}_A(X)$.

(2) If α and β are expressions in $\mathrm{REG}_A(X)$, then so are $\alpha \cup \beta$ (union), $\alpha \cdot \beta$ (concatenation), α^* (star), and $\bar{\alpha}$ (complementation).

Note that the constant languages in the definition of $\mathrm{REG}_A(X)$ are not required to be regular; they can be completely arbitrary.

In order to formulate different language equation problems, one can restrict α to some subset of $\mathrm{REG}_A(X)$. For related work, we refer to [8] for classical language equations; to [2, 3, 5] for equations with regular constants and with union, concatenation from the left by a constant, and complementation as operators; to [4] for equations with regular constants and union, concatenation from the left by a constant, and star as operators; and to [6] for equations where the constants are over a one-letter alphabet and the operators are union, concatenation, and star. This last paper is closest to the type of equations that we study here; it shows that any system of such equations has a regular solution if all constants are regular (and in many cases, even if they are not) and gives a constructive method of determining it. In fact, the order is reversed: it is first shown that certain expressions are solutions and then verified that these are regular based on the form of the expressions.

Adding complementation substantially increases the difficulty of dealing with a equation. While the classical theory of equations has been known for several decades (see [8]), it took the introduction of boolean automata in [2] to handle the addition of complementation to equations where concatenation is restricted (left concatenation) and the star is not present. In more general cases, in particular if concatenation is unrestricted, there are very few methods that allow one to deal at all with equations of this type. An exception is the derivative method [1] that permits one to determine whether a word is in a solution provided a condition similar to our λ-property (see below) holds. However, this method is not useful if one is interested in a representation of all words in a solution. Since one is usually interested in a closed solution, especially if the solution is regular, one tends to favor equations with regular solutions; all the quoted papers do this. It also follows that in the presence of unrestricted concatenation this will be impossible to achieve, even without complementation, unless one restricts the alphabet to contain only one letter, henceforth denoted by a. This was done in [6]; the star operator was also permitted there since this turned out to be necessary in order to solve equations with unrestricted concatenation only. In that paper a method was presented for obtaining closed regular expressions for the solutions of systems of such equations.

The present paper studies the most general equations over a one-letter alphabet, namely those where the operators are union, unrestricted concatenation, star, and

complementation (i.e., all the operators that usually occur in the definition of regular expressions and under all of which the regular languages are closed). We first define what we call the λ-property of expressions and use it to show existence and uniqueness of solutions. Then we introduce a substitution procedure that we use to obtain regular solutions of equations where the constants are regular. This procedure works even though complementation does not possess the substitution property. This raises several important issues; in particular, that of convergence and the derivation of regular expressions. In the second part of the paper we show that there exist equations over the alphabet $\{a\}$ with complementation and regular constants whose solutions are not context-free. This result has several interesting implications.

2. The λ-property

Consider the following property of expressions $\alpha(X)$, called λ-property, defined by induction on the structure of α:

(a) Basis:

- If L is a language over the alphabet A, then L has the λ-property.
- If X is a variable, then X does not have the λ-property.

(b) Induction step:

- $\alpha \cup \beta$ has the λ-property iff both α and β have the λ-property.
- $\bar{\alpha}$ has the λ-property iff α has the λ-property.
- α^* has the λ-property iff α has the λ-property.
- $\alpha \cdot \beta$ has the λ-property iff α has the λ-property and β has the λ-property when $\delta(\alpha) = \lambda$.

This requires defining the function δ which is by induction on the structure of the expression α:

(a) Basis:

- $\delta(L) = \emptyset$ if $\lambda \notin L$ and $\delta(L) = \lambda$ if $\lambda \in L$.
- $\delta(X) = ?$.

(b) Induction step: Assume the values of $\delta(\alpha)$ and $\delta(\beta)$ are known. We have to define the union, complement, star, and concatenation.

$\delta(\alpha \cup \beta)$ is given by the following table:

	$\delta(\beta)$			
$\delta(\alpha)$ \	λ	$\emptyset$	$?$	$\overline{?}$
λ	λ	λ	λ	λ
$\emptyset$	λ	$\emptyset$	$?$	$\overline{?}$
$?$	λ	$?$	$?$	λ
$\overline{?}$	λ	$\overline{?}$	λ	$\overline{?}$

$\delta(\bar{\alpha})$ is given by the following table:

$\delta(\alpha)$	$\delta(\bar{\alpha})$
λ	$\emptyset$
$\emptyset$	λ
?	$\bar{?}$
$\bar{?}$	?

$\delta(\alpha^\star) = \lambda$ for all expressions α.
$\delta(\alpha \bullet \beta)$ is given by the following table:

$\delta(\alpha)$ \ $\delta(\beta)$	λ	$\emptyset$	?	$\bar{?}$
λ	λ	$\emptyset$	?	$\bar{?}$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
?	?	$\emptyset$	?	$\emptyset$
$\bar{?}$	$\bar{?}$	$\emptyset$	$\emptyset$	$\bar{?}$

The goal in defining $\delta(\alpha)$ is to capture whether the expression α contains the empty word λ. In the absence of variables, δ is the standard function that does just that for regular expressions [1]. The values ? and $\bar{?}$ represent two different kinds of absence of information which are introduced by the variable X and its complement. This is particularly clear in union where the union of X and $\bar{X}$ will always contain λ. The symbols ? and $\bar{?}$ follow the usual rules for set operations; thus, complementing ? twice yields ?.

As illustration, consider $\alpha = ((aa)^\star \cap X)\bar{X}$. According to our definitions (and using deMorgan's laws) it follows that $\delta(\alpha) = (\lambda \cap ?)\ \bar{?} = ?\ \bar{?} = \emptyset$. Furthermore, $\delta((aa)^\star X \cup \bar{\alpha}\bar{X}) = \lambda ? \cup \bar{\emptyset}\ \bar{?} = ? \cup \lambda\ \bar{?} = ? \cup \bar{?} = \lambda$.

These definitions can be extended to equations in several variables; however, we must distinguish between the different values ?. Thus, if we have variables X and Y, we must also have $?_X$ and $\bar{?}_Y$. Note that the uncertainty values for different variables are unrelated; thus, even though $?_X \cup \bar{?}_X$ is equal to λ, $?_X \cup \bar{?}_Y$ is not. On the other hand, the values λ and $\emptyset$ are independent of any variables. In this paper, we will deal with equations in the one variable X only. Also, since all languages are over a one-letter alphabet, they are commutative ($u \bullet v = v \bullet u$ for all words u and v and $L \bullet M = M \bullet L$ for all languages L and M). However, the λ-property and the δ-function are defined independently of commutativity.

The idea behind the λ-property is to guarantee that we do not have equations where X might be defined in terms of itself. This is an old idea; it is precisely the motivation for requiring in the classical equation [8]

$$X = L \bullet X \cup M$$

that $\lambda \notin L$. A similar notion of λ-property with the same purpose (and even with the same name) was defined in [5]. The only difference here is that the definition of the δ-function is substantially more complicated.

3. Existence and uniqueness of solutions

In this section we show that any equation $X = \alpha$ where α is a one-letter alphabet and its operators are union, concatenation, complementation and star has always a solution and that this solution is unique, provided that α has the λ-property. Before we do this, we must define another function

$$\mathrm{PROJ}(\alpha, S, w),$$

which denotes the projection of the word w on the expression $\alpha(X)$; it is defined only if the word w is contained in a solution S of the equation $X = \alpha(X)$ and this solution S must be a parameter of the function PROJ. It maps syntactic components (subexpressions) $\beta(X)$ of $\alpha(X)$ to subwords v of w such that

$$v \in \beta(S).$$

If there are several such mappings, $\mathrm{PROJ}(\alpha, S, w)$ contains all of them, with the exception that for $\beta = \gamma^\star$, the empty subword may be selected at most once to be mapped to γ. Under this restriction, the number of mappings in $\mathrm{PROJ}(\alpha, S, w)$, for any α, S, and w as appropriate, is always finite.

Consider the expression $\alpha(X) = ((aa)^\star \cap X)^\star \bullet a \bullet \bar{X}$. It follows that $\delta(\alpha) = \emptyset$ because $\delta((aa)^\star) = \lambda$ and $\delta(X) = ?$, and consequently $\delta((aa)^\star \cap X) = ?$ and $\delta(((aa)^\star \cap X)^\star) = \lambda$. We can verify that $S = a(aa)^\star$ is a solution of the equation $X = \alpha(X)$ (by direct substitution). Let us determine

$$\mathrm{PROJ}(\alpha(X), a(aa)^\star, a^7).$$

Note that for $\cup$ (union) exactly one branch in the tree must be exercised, for $\bullet$ (concatenation) and $\cap$ (intersection) both branches must be used, and for * (star) zero or more iterations must be indicated. The unique syntax tree for $\mathrm{PROJ}(A, S, a^7)$ is given in Fig. 1.

We note that for subexpressions of the form $\gamma^\star$ where γ contains λ it is possible to obtain i iterations of γ each with λ for arbitrarily many values of i. However, if we explicitly prohibit this, the number of elements in $\mathrm{PROJ}(\alpha, S, w)$ is finite:

Proposition 3.1. *If for any subexpression β of α which is of the form*

$$\beta = \gamma^\star,$$

the empty subword of w may be mapped at most once to β, the number of mappings in $\mathrm{PROJ}(\alpha, S, w)$ is finite for any expression α, any solution S of the equation $X = \alpha(X)$, and any word $w \in S$.

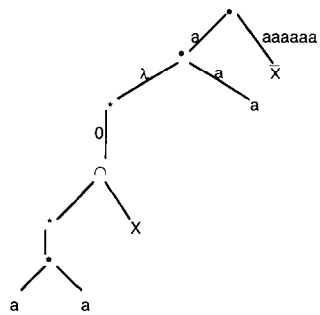

Fig. 1. PROJ($\alpha(X), S, aaaaaaa$).

Proof. The proof follows from the fact that w is of finite length and if the empty word cannot be arbitrarily repeated, there are only finitely many ways of decomposing w into subwords. □

PROJ(α, S, w) may contain more than one tree: Let $\alpha(X) = aX \cup aaX \cup a$. One knows from classical language equations that there is a unique solution S,

$$S = aa^{\star}.$$

Clearly, a^3 is in S and has exactly two different mappings, one that decomposes it as $a \bullet a \bullet a$ where the first two subwords (a's) come from the subexpression aX, and one that decomposes it as $aa \bullet a$ with the first subword (aa) coming from the subexpression aaX.

A more complicated example is given by the equation

$$X = \bar{X}X \cup a,$$

which has a solution $S = a \cup a^3a^{\star}$. Note that $\delta(\bar{X} \bullet X \cup a) = \emptyset$; because of Theorem 3.2, this solution is therefore unique. The word a has two mappings; one is the obvious one mapping a to a, the other maps a to $\bar{X} \bullet X$ which is valid since $\lambda \in \bar{X}$ and $a \in X$.

Theorem 3.2. *Consider the equation $X = \alpha(X)$ over the one-letter alphabet $\{a\}$. If α has the λ-property the equation has exactly one solution.*

Proof. First we show uniqueness. Let S_1, S_2 be two solutions, $S_1 \neq S_2$. Let w be a shortest word in the symmetric difference $S_1 \Delta S_2$; w.l.o.g. $w \in S_1 - S_2$. Consider PROJ(α, S_1, w). It is clear that w must have a nonempty subword mapped to X by PROJ(α, S, w) since otherwise it would be mapped only to constant languages and therefore would have to be in every solution. This means in particular that $w \neq \lambda$. Because α has the λ-property, there is at least one subword of nonzero length in w that must be mapped to a constant language in α; the only exception is a subexpression $\bar{X} \bullet X$, in which case one of the two must contain λ, the other must not. This achieves

the same reduction in length that is necessary. It follows that any subword v mapped to X (occurring in a subexpression other than $\bar{X} \cdot X$) is of length less than $|w|$. Since w is shortest with respect to $S_1 - S_2$, all v's must be in S_2 and this in turn implies that $w \in S_2$ as well, using the mapping in $\mathrm{PROJ}(\alpha, S_1, w)$ as a construction rule for the word w in S_2. This is in contradiction to $w \notin S_2$; hence, any solution must be unique.

Existence of a solution follows immediately from the following proposition.

Proposition 3.3. *If $\alpha(x)$ has the λ-property, the derivative α_w of $\alpha(X)$ with respect to any word w is uniquely defined and $\delta(\alpha_w) \in \{\emptyset, \lambda\}$.*

Proof. This proposition follows by an easy but somewhat tedious application of derivatives [1] to our definition of δ. Defining $S = \{w \in A^\star \mid \delta(\alpha_w) = \lambda\}$, one shows that S is a solution of the equation. □

Although the theorem establishes existence and uniqueness of the solution of an equation $X = \alpha(X)$, where α has the λ-property, it does not provide much help in finding this solution. Complementation is generally considered "hard"; for example, the substitution property does not hold. The equation $X = a\bar{X}$ over $\{a\}$ has the unique solution $a(aa)^\star$, but if one substitutes the language $b^\star$ for the letter a to get a solution of the equation $X = b^\star\bar{X}$ over $\{b\}$, it turns out that $b^\star(b^\star b^\star)^\star = b^\star$ is not a solution of the equation; in fact $X = b^\star\bar{X}$ does not have any solution [5]. More importantly, there does not seem to exist a way of reducing levels of complementation in the presence of variables. In [6] it was shown how to reduce the level of stars in any expression with union, concatenation, and star to 1, using the formula $(BC^\star)^\star = BB^\star C^\star \cup \lambda$ for all languages B, C, over $\{a\}$. There does not appear to be any way to do this with complementation. This is because concatenation does not distribute over complementation. As an example, consider the attempt to simplify the expression α over the one-letter alphabet $\{a\}$,

$$\alpha = \overline{a^\star\bar{B}}.$$

It follows that

$$\alpha = \begin{cases} \emptyset & \text{if } \lambda \notin B, \\ a^\star & \text{if } B = a^\star, \\ a^{0..t-1} & \text{if } \lambda \in B \text{ and } a^t \text{ is the shortest nonempty word not in } B. \end{cases}$$

($a^{m..n}$ denotes the set of all words in $a^\star$ of length at least m and at most n.) Thus, knowledge of B is required to determine $C\bar{B}$ even for very simple C. It is unlikely that a formula with only one level of complementation can be found, using only syntactic transformations; this is even more so for expressions containing the concatenation of two variables, such as

$$\overline{X\bar{Y}}.$$

4. The substitution procedure

Let the equation $X = \alpha(X)$ be given; we can view α also as a function of the language variable X. Any regular language R over the alphabet $\{a\}$ has a normal form, given by

$$R = F \cup G \bullet (a^t)^\star,$$

where F and G are finite languages,

$$F = \{a^{f_1}, \ldots, a^{f_m}\}, \qquad G = \{a^{g_1}, \ldots, a^{g_n}\},$$

$t \geqslant 1$, $m \geqslant 0$, $n \geqslant 0$ and $0 = f_0 \leqslant f_1 < \cdots < f_m < g_1 < \cdots < g_n$, with $g_n - g_1 < t$ and $f_m + t \leqslant g_n$, obtained from the reduced deterministic automaton for R. Therefore, this normal form is unique.

4.1. Description of the substitution procedure $SP(\alpha(X))$

(1) Choose an initial guess S_0; this initial guess should be regular and in normal form.

(2) For $i = 1, 2, \ldots$, compute $S_i := \alpha(S_{i-1})$ and determine its normal form. For each obtained S_i, compare it to the previously computed S_j's; if S_i is equal to one S_h, stop.

From now on, we assume that α possesses the λ-property; this guarantees us the existence and uniqueness of a solution. This substitution procedure is a variant of fix point methods (see, for instance [9]); note however that because of the presence of complementation, monotonicity does not hold.

(1) *The initial guess.* It is advisable to start with $S_0 := \delta(\alpha)$; this ensures that one does not start with the complement of the eventual solution, since $\delta(\alpha)$ will always be contained in the solution. If one does not choose $S_0 = \delta(\alpha)$ at worst a few additional steps will be necessary.

(2) *The question of convergence.* Several questions relate to convergence. First we consider the problem of S_i being equal to S_h for $i < h$. If $i = h + 1$, then we have the ideal situation; Since here $S_i = \alpha(S_i - 1)$, we have obtained a fix point of our equation; in other words, S_i is a solution of the equation, and by Theorem 3.2, it is the unique solution. If $i < h + 1$, then we have an oscillation, since from that point onwards all the languages between S_h and S_i will repeat cyclicly: $S_t = S_i$ for every $t = i + (i - h) \bullet k$ for all $i \geqslant 0$. We claim that the case $h - i \geqslant 2$ cannot occur if the λ-property holds. The proof of this is a consequence of a proposition that we formulate and prove below. Finally, there is the possibility that $S_i = S_h$ for some $h < i$ never occurs. This is definitively possible (see the examples below) and in this case there is no convergence.

Proposition 4.1. *Let α possess the λ-property, and (by Theorem 3.2) let S be the unique solution of the equation $X = \alpha(X)$. If the procedure $SP(\alpha(X))$ is started with $S_0 := \delta(\alpha)$,*

we know after i steps of the procedure for any word of length at most i whether it is in S or not. In other words, S_i contains all words of S of length at most i, for all $i \geqslant 0$,

$$S_i \cap \{a^0, \ldots, a^i\} = S \cap \{a^0, \ldots, a^i\}.$$

Proof. By induction on i. If $i = 0$, then by assumption $S_0 = \delta(\alpha)$, and either $\lambda \in S$, then $\lambda \in \delta(\alpha)$, or $\lambda \notin S$ then $\lambda \notin \delta(\alpha)$. In both cases the claim holds. Now assume the claim holds for i; we have to show it also holds for $i + 1$. This follows again by the use of PROJ. The key idea is that because of the λ-property, any word in S_{i+1} must be mapped to subexpressions of $\alpha(X)$ and all those subwords mapped to X are words in S_i. Since by inductive hypothesis S_i satisfies the claim, S_{i+1} does now too. □

This result allows us now to define convergence more precisely. By convergence we mean that the "relevant" portion of S_i (all words of length at most i) is contained in the "relevant" portion of S_j (all words of length at most j) for all $j > i$. Note that because of the presence of complementation the usual argument of iterated substitution, namely that the words in S_i are shorter than those in S_{i+1}, does not work.

Now we show that oscillations cannot occur. Let $S_h, S_{h+1}, \ldots, S_{i-1}$ be the repeated sequence, i.e., $S_k = S_{k+(i-h)}$ with $h - i \geqslant 2$, for all $k \geqslant h$. In particular, $S_h \neq S_{h+1}$. Let w be the shortest word in the symmetric difference $S_h \Delta S_{h+1}$ and let t be the smallest number $> |w|$ that is equal to $h + (i - h) \cdot c$ for some integer $c \geqslant 0$. Consider S_t: By the above $w \in S_t \Delta S_{t+1}$, but by Proposition 4.1 we know that if $w \in S_t$, then it is permanently there, i.e., $w \in S_p$ for all $p \geqslant t$; in particular for $p = t + 1$. This yields a contradiction to the assumption $w \in S_t \Delta S_{t+1}$. Therefore, oscillations cannot occur.

Example. Consider the equation $X = \alpha(X)$ where

$$\alpha = a \cdot \bar{X} \cdot \bar{X};$$

we use the procedure SP to determine its solution. First we verify that the expression α possesses the λ-property. Then we choose $\delta(\alpha)$ as initial guess.

$$S_0 = \emptyset.$$

The next few S_i's follow:

$$S_1 = a \cdot a^\star \cdot a^\star = a \cdot a^\star,$$

$$S_2 = a \cdot \lambda \cdot \lambda = a,$$

$$S_3 = a \cdot (\lambda \cup aaa^\star) \cdot (\lambda \cup aaa^\star) = a \cup a^3 a^\star,$$

$$S_4 = a \cdot (\lambda \cup a^2) \cdot (\lambda \cup a^2) = a \cup a^3 \cup a^5,$$

$$S_5 = a \cdot (\lambda \cup a^2 \cup a^4 \cup a^6 a^\star) \cdot (\lambda \cup a^2 \cup a^4 \cup a^6 a^\star)$$

$$= a \cup a^3 \cup a^5 \cup a^7 a^\star.$$

At this point it should be quite obvious that this process will never end. Therefore, one may venture to guess that only strings of odd length will be produced; to verify this guess one substitutes the guess

$$G = a(aa)^\star$$

into the equation; this gives

$$a \cdot (aa)^\star(aa)^\star = a(aa)^\star = G.$$

Therefore, our guess G is in fact the unique solution of the equation.

Example. The equation

$$X = a^2 \cdot X \cdot \bar{X} \cup a^i$$

has the following set of solutions:
For $i = 0$, the unique solution is $\lambda \cup a^{3..4} \cup a^6a^\star$ (direct);
for $i = 1$, it is $a(aa)^\star$ (guess), and
for all $i \geqslant 2$, it is $a^{i+2}a^\star \cup a^i$ (direct).

Example. Consider the equation

$$X = a \cdot X \cdot \bar{X} \cdot \bar{X} \cup \lambda;$$

it can be verified that it has the solution $\lambda \cup a^3a^\star$ which is obtained directly in S_1 (two steps of SP) if we start with $S_0 = \delta(\alpha)$. Suppose we start SP instead with the complement of the solution

$$S_0 = a^{1..2}.$$

We get

$$S_1 = \lambda \cup a^{2..3} \cup a^5a^\star,$$

$$S_2 = \lambda \cup a^3 \cup a^{5..6} \cup a^8a^\star,$$

$$S_3 = \lambda \cup a^3a^\star.$$

As one can see, two more steps are required but the direct solution eventually emerges.

In general, we can formulate the following corollary to Proposition 4.1 which essentially states that the initial guess does not affect the containment of words of length at most i in S_i:

Corollary 4.2. *Let α possess the λ-property, and let S be the unique solution of the equation $X = \alpha(X)$. Let M be an arbitrary language over $\{a\}$. If the procedure* SP($\alpha(X)$) *is started with $S_0 := M$, we know after i steps of the procedure for any word of length at most i whether it is in S or not. In other words, S_i contains all words of S of length at most i, for all $i \geqslant 0$,*

$$S_i \cap \{a^0, \ldots, a^i\} = S \cap \{a^0, \ldots, a^i\}.$$

Proof. The proof is a simple modification of that of Proposition 4.1 and uses the fact that because of the λ-property, $\lambda \in S$ iff $\lambda \in \alpha(M)$ for any language M. □

Example. Consider the equation $X = \alpha(X)$, with α given by

$$\alpha = a^5 \cdot X \cdot [(a^7 \cup a^{12}) \cdot X^2 \cdot \bar{X} \cap (a^7 \cdot X \cdot \overline{(a^3)^\star \cdot X})^\star] \cup a^8 \cdot X.$$

One first verifies that α has the λ-property and that $\delta(\alpha) = \emptyset$. Then one computes (we suppress the tedious intermediate results):

$$S_0 = \emptyset,$$

$$S_1 = a^8 a^\star,$$

$$S_2 = a^{8..15} \cup a^{36} a^\star,$$

$$S_3 = a^{8..15} \cup a^{24} a^\star,$$

$$S_4 = a^{8..15} \cup a^{24..31} \cup a^{36} a^\star.$$

At this point one may guess that the solution is $G = a^{8..15} \cdot (a^{16})^\star$. Substituting G into α gives the following language:

$$a^{8..15} \cup a^{24..31} \cup a^{36} a^\star,$$

which is not equal to G. Therefore, continuing the procedure, one obtains

$$S_5 = a^{8..15} \cup a^{24..31} \cup a^{36} a^\star$$

and therefore $S_5 = S_4$. Thus, the unique solution S of the equation is S_4.

At this point one might well conjecture that all such equations have regular solutions. That this is not the case is shown in the next section.

5. Non-regularity of solutions

Recall that $\mathrm{REG}_A(X)$ was defined as the set of all (extended) regular expressions in X over A. In particular, nothing was implied about the constant languages in these expressions. An equation $X = \alpha(X)$ will be called *regular* if all the constant languages occurring in α are regular. No statement is made about the type of the solution, if one exists. In order to simplify our notation, we abbreviate as before $a^{m..n} = \{a^m, a^{m+1}, \ldots, a^n\}$ and $\alpha^2 = \alpha\alpha$ for any expression α.

Theorem 5.1. *There exist regular language equations in one variable over the alphabet $A = \{a\}$ using concatenation and complementation only which have non-context-free solutions.*

Proof. Consider the equation

$$X = a\overline{\overline{X^2}^2}^2.$$

First, we note that by Theorem 3.1, this equation has a unique solution, because the expression on the right has the λ-property. Then we observe that the following language is a solution:

$$S = \bigcup_{i \geqslant 0} a^{2^{3i}..2^{3i+2}-1}.$$

This can be verified by direct substitution of S into the equation:

$$\overline{S} = \lambda \cup \bigcup_{i \geqslant 0} a^{2^{3i+2}..2^{3i+3}-1},$$

$$\overline{S}^2 = \lambda \cup \bigcup_{i \geqslant 0} a^{2^{3i+2}..2^{3i+4}-2},$$

$$\overline{\overline{S}^2} = \bigcup_{i \geqslant 0} a^{2^{3i+1}-1..2^{3i+2}-1},$$

$$\overline{\overline{S}^2}^2 = \bigcup_{i \geqslant 0} a^{2^{3i+1}..2^{3i+3}-2},$$

$$\overline{\overline{\overline{S}^2}^2} = \bigcup_{i \geqslant 0} a^{2^{3i}-1..2^{3i+1}-1},$$

$$\overline{\overline{\overline{S}^2}^2}^2 = \bigcup_{i \geqslant 0} a^{2^{3i}-1..2^{3i+2}-2},$$

and from this the claim follows.

We now show that S is not regular; more specifically, we claim that for any integer $M \geqslant 1$, there exists an integer M_0 such that

$$a^{M_0..M_0+M-1} \cap S = \emptyset.$$

Indeed, there are arbitrarily large gaps between successive powers of 2. In our case, the gap between $a^{2^{3i+2}}$ and $a^{2^{3i+3}}$ is of size 2^{3i+2}. Thus, for any given M, M_0 is equal to the smallest number of the form 2^{3i+2} not less than M,

$$M_0 = 2^{3\lceil(\log_2 M - 2)/3\rceil + 2},$$

Therefore, S cannot be regular, and thus S cannot be context-free either. In fact, S cannot even lie in the closure of CFL under all boolean operations, concatenation and star. □

Remark. The solution S was obtained by applying the procedure SP to the above equation. After several steps, a pattern emerged suggesting the particular form of S. Even though SP was designed to determine regular solutions only, it proved to be applicable to this equation as well. We consider it unlikely that such a result could be obtained in another way.

Observation. The above equation is the simplest with a nonregular solution, in the following sense: Dropping one or more complementation or squaring operations in

the regular expression on the right yields an equation whose (unique) solution is regular. This can be verified directly.

Theorem 5.2. *Consider the language equation $X = \alpha$ over the alphabet $\{a\}$ with regular constant languages, where the operations are union, concatenation, star, and complementation. If this equation has any solution, then there always exists a context-sensitive one.*

Proof. The proof follows by constructing a linear bounded automaton which simulates exactly the construction of PROJ(α, L, w), where L ranges over all those subsets of $a^{\leq |w|}$ which contain w. This clearly can be done in linearly bounded space since each subset L can be recorded as a binary string, with a 1 in position i iff $a^i \in L$. This is based on the observation that a string of length n can be in a solution only on the basis of whether other strings of length *at most* n are in the solution, regardless of whether complementation is involved or not. In other words, whether a string of length n is or is not in a solution is independent of whether a longer string is in the solution. This is the justification for trying out all possible languages L in PROJ(α, L, w) even though none of them may in fact be (part of) a solution. □

Corollary 5.3. *There exist regular language equations over an arbitrary alphabet using only concatenation and complementation which have solutions that are not contained in the closure of CFL under boolean operations, concatenation, and star.*

Proof. Consider the same equation as in the proof of Theorem 5.2, except that the alphabet is now arbitrary, say $\{a,b, \ldots\}$. Let S, T be the solutions of this equation for the alphabets $\{a\}$, $\{a,b, \ldots\}$, respectively. It follows that $S = T \cap a^*$, and this implies the statement of the corollary. □

6. Conclusion and open problems

Certain language equations have always provided a means to start with one class of languages and get solutions in a more general class. Usually, concatenation is implicated in such situations. Examples are given by the equations derived from context-free grammars (e.g., $X = aXb \cup \lambda$ derived from the context-free productions $X \rightarrow aXb \mid \lambda$) and the star equations studied in [4]. However, in this way one could never get more than CFL when starting from regular languages. This paper has shown that adding complementation to the canon of permitted operations will result in equations whose solutions are no longer even context-free nor do they lie in the closure of CFL under the standard operations. This appears to be the first demonstration that the notion of language equations with the standard operations can be used to generate non-context-free languages.

We raise the following fundamental question: Is it decidable whether a given equation $X = \alpha(X)$ has a regular solution? The problem is clearly r. e.; the procedure SP proves this. However, we know that SP does not terminate in general. Thus, a question whose positive answer would imply the decidability of this problem is whether one can give an upper bound on the number of steps of SP which when exceeded guarantees that the solution is not regular. This question in turn can be decomposed into two subproblems.

(A) Given a reduced automaton $\mathbb{R}$ with n states accepting a regular language over $\{a\}$, for how many words must one know acceptance or rejection before one can determine the automaton itself?

This question is made easier since any guess can be conclusively verified or rejected by substitution. Thus, one can certainly determine $\mathbb{R}$ if one knows about all words a^i, $i = 0, 1, \ldots, 2n$, whether or not they are accepted by $\mathbb{R}$.

(B) Given an equation $X = \alpha(X)$ with a regular solution S, is there an upper bound, in terms of $\alpha(X)$ only, on the number of states of the reduced automaton accepting S?

This question appears to be very difficult. In addition, in view of the results in [7] on complementation in regular expressions, it is not inconceivable that such a bound, if it does exist, is nonelementary.

References

[1] J.A. Brzozowski, Canonical regular expressions and minimal state graphs for definite events, in: *Mathematical Theory of Automata*, Symposia Series **12** (Polytech. Institute of Brookyln, Brooklyn, 1963).

[2] J.A. Brzozowski and E.L. Leiss, On equations for regular languages, finite automata and sequential networks, *Theoret. Comput. Sci.* **10** (1980) 19–35.

[3] E.L. Leiss, On generalized language equations, *Theoret. Comput. Sci.* **14** (1981) 63–77.

[4] E.L. Leiss, On solving star equations, *Theoret. Comput. Sci.* **39** (1985) 327–332.

[5] E.L. Leiss, Generalized language equations with multiple solutions, *Theoret. Comput. Sci.* **44** (1986) 155–174.

[6] E.L. Leiss, Language equations over a one-letter alphabet with union, concatenation, and star: a complete theory, *Theoret. Comput. Sci.* **131** (1994) 311–330.

[7] A.R. Meyer and L. Stockmeyer, Nonelementary word problems in automata and logic, in: *Proc. AMS Symposium on Complexity of Computation*, 1973.

[8] A. Salomaa, *Theory of Automata* (Pergamon Press, Oxford, 1969).

[9] A. Salomaa and A. Soitola, *Automata Theoretic Aspects of Formal Power Series* (Springer, Berlin, 1986).